

composer2d.py

```
1  from utils import *
2  from shape2d import Vertex, Shape2D
3
4  def Line(_position : Vertex = Vertex(0, 0, 0), _edge : float = 1, _color : tuple[float] =
  None, _rotation : tuple[float, str] = None):
5      """
6      Initialize 2D Line object .
7      """
8      half_edge = _edge / 2
9      VERTICES = [
10         Vertex(
11             _position.x,
12             _position.y,
13             _position.z - half_edge,
14         ),
15         Vertex(
16             _position.x,
17             _position.y,
18             _position.z + half_edge,
19         ),
20     ]
21
22     return Shape2D(_position, VERTICES, _color, GL_LINES)
23
24
25 def Axes():
26     """
27     Initialize 2D Axes object .
28     """
29     half_edge = 4
30     VERTICES = [
31         Vertex(
32             0,
33             0,
34             0,
35         ),
36         Vertex(
37             0,
38             0,
39             -half_edge,
40         ),
41
42         Vertex(
43             0,
44             0,
45             0,
46         ),
47         Vertex(
48             0,
49             0,
50             half_edge,
51         ),
52
53         Vertex(
54             0,
55             0,
```

```

56         0,
57     ),
58     Vertex(
59         0,
60         -half_edge,
61         0,
62     ),
63
64     Vertex(
65         0,
66         0,
67         0,
68     ),
69     Vertex(
70         0,
71         half_edge,
72         0,
73     ),
74
75     Vertex(
76         0,
77         0,
78         0,
79     ),
80     Vertex(
81         -half_edge,
82         0,
83         0,
84     ),
85
86     Vertex(
87         0,
88         0,
89         0,
90     ),
91     Vertex(
92         half_edge,
93         0,
94         0,
95     ),
96
97 ]
98 return Shape2D(Vertex(0, 0, 0), VERTICES, COLORS['green'], GL_LINES)
99
100
101 def Net(_size : int = 4):
102     """
103     Initialize 2D Net (every 1 unit = line) object .
104     """
105     len = _size * 2
106     VERTICES = []
107     for z in range(-_size, _size+1):
108         for a in range(-_size, _size+1):
109             VERTICES.append(Vertex(a, -_size, z))
110             VERTICES.append(Vertex(a, +_size, z))
111
112             VERTICES.append(Vertex(-_size, a, z))
113             VERTICES.append(Vertex(+_size, a, z))
114
115     for y in range(-_size, _size+1):

```

```

116         for x in range(-_size, _size+1):
117             VERTICES.append(Vertex(x, y, -_size))
118             VERTICES.append(Vertex(x, y, +_size))
119
120     return Shape2D(Vertex(0, 0, 0), VERTICES, COLORS['g_cage'], GL_LINES)
121
122
123 def Cage(_size : int = 4):
124     """
125     Initialize 2D Cage (Borders) object .
126     """
127     VERTICES = [
128         Vertex(
129             -_size,
130             -_size,
131             -_size,
132         ),
133         Vertex(
134             +_size,
135             -_size,
136             -_size,
137         ),
138         Vertex(
139             -_size,
140             -_size,
141             -_size,
142         ),
143         Vertex(
144             -_size,
145             +_size,
146             -_size,
147         ),
148         Vertex(
149             -_size,
150             -_size,
151             -_size,
152         ),
153         Vertex(
154             -_size,
155             -_size,
156             +_size,
157         ),
158
159         Vertex(
160             -_size,
161             +_size,
162             -_size,
163         ),
164         Vertex(
165             -_size,
166             +_size,
167             +_size,
168         ),
169         Vertex(
170             -_size,
171             +_size,
172             -_size,
173         ),
174         Vertex(
175             +_size,

```

```
176         +_size,
177         -_size,
178     ),
179
180     Vertex(
181         +_size,
182         +_size,
183         +_size,
184     ),
185     Vertex(
186         -_size,
187         +_size,
188         +_size,
189     ),
190     Vertex(
191         +_size,
192         +_size,
193         +_size,
194     ),
195     Vertex(
196         +_size,
197         -_size,
198         +_size,
199     ),
200     Vertex(
201         +_size,
202         +_size,
203         +_size,
204     ),
205     Vertex(
206         +_size,
207         +_size,
208         -_size,
209     ),
210
211     Vertex(
212         +_size,
213         -_size,
214         +_size,
215     ),
216     Vertex(
217         -_size,
218         -_size,
219         +_size,
220     ),
221     Vertex(
222         +_size,
223         -_size,
224         +_size,
225     ),
226     Vertex(
227         +_size,
228         -_size,
229         -_size,
230     ),
231
232     Vertex(
233         -_size,
234         -_size,
235         +_size,
```

```

236         ),
237         Vertex(
238             -_size,
239             +_size,
240             +_size,
241         ),
242         Vertex(
243             +_size,
244             -_size,
245             -_size,
246         ),
247         Vertex(
248             +_size,
249             +_size,
250             -_size,
251         ),
252     ]
253
254
255     return Shape2D(Vertex(0, 0, 0), VERTICES, COLORS['g_cage'], GL_LINES)
256
257
258 def Square(_position : Vertex = Vertex(0, 0, 0), _edge : float = 1, _color : tuple[float]
= None, _rotation : tuple[float, str] = None):
259     """
260     Initialize 2D Square object .
261     """
262     half_edge = _edge / 2
263     VERTICES = [
264         Vertex(
265             _position.x - half_edge,
266             _position.y - half_edge,
267             _position.z,
268         ),
269         Vertex(
270             _position.x + half_edge,
271             _position.y - half_edge,
272             _position.z,
273         ),
274         Vertex(
275             _position.x + half_edge,
276             _position.y + half_edge,
277             _position.z,
278         ),
279         Vertex(
280             _position.x - half_edge,
281             _position.y + half_edge,
282             _position.z,
283         )
284     ]
285     square = Shape2D(_position, VERTICES, _color)
286     if _rotation is not None:
287         square.rotate(*_rotation)
288     return square
289
290
291 def Cirlce(_position : Vertex = Vertex(0, 0, 0), _radius : float = 1, _n_sides : int =
100, _color : tuple[float] = None):
292     """
293     Initialize 2D Cirlce object .

```

```
294     """
295     angle = 2 * pi / _n_sides
296     VERTICES = []
297     for i in range(_n_sides):
298         vertex = Vertex(
299             _position.x,
300             _position.y + _radius,
301             _position.z,
302         )
303         vertex.rotate(_position, i * angle, 'Ro_z')
304         VERTICES.append(vertex)
305     return Shape2D(_position, VERTICES, _color)
306
307
308
```